INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

Laboratoire CITI / INRIA

Research internship

Title: **Real-time Linux Programming on the Raspberry Pi**

Location: CITI Lab, INSA-Lyon/INRIA. Villeurbanne, France.

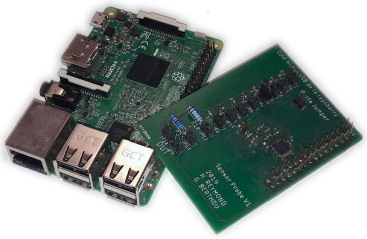
Keywords: Bare-Metal Programming, Linux kernel, ARM, RPi 3, SPI, DMA

Contact: `guillaume.salagnac@insa-lyon.fr` — `http://perso.citi-lab.fr/gsalagnac`

Context

The general context for this project is the Internet of Things, i.e. a vision where we are gradually surrounded by intelligent, connected objects. More precisely, we are interested in tiny devices with limited CPU, memory, and power resources. We study the impact of certain embedded operating system features on the power consumption of the device. This requires the ability to perform fine-grained current measurements over an extended period of time, which is not possible with ordinary instruments.

In previous work, we have already designed and built a power metering system dedicated to ultra low power devices. Our prototype is implemented as a shield daughterboard for the Raspberry Pi 3 (cf picture) with accompanying bare-metal software to drive the experiment.



The actual current measurement is performed using the classical method: a small shunt resistor translates current into a voltage, which is amplified and then digitized by an ADC. The main role of the Raspberry Pi is to read the ADC samples over SPI. In our prototype, the RPi doesn't process the data any further, though. Instead, it forwards the raw data stream to a PC via serial/USB.

The reason for this is real-time performance. For the measures to be meaningful, we need to control the timing very precisely. This is why we don't run the usual Linux software stack, but an ad-hoc RPi firmware[1] instead. This program runs on two CPU cores in parallel (out of 4)

Goals

The objective of this internship is to port all of the features of the bare-metal firmware into a proper Linux system. Instead of having to connect the RPi to a PC via USB and run all data analysis tools there, it would be more efficient to work directly on the RPi. In addition, it would allow for more advanced experimentation scenarios.

The main function of the RPi is to poll the ADC with a fixed frequency and minimal jitter. In a preliminary effort, we found out that the simple idea of writing a userland program to poll the SPI driver does not meet the required performance (excessive syscall overhead, uncertainty in scheduling, etc). Instead, we will probably have to push the main loop in the lower software layers, for instance into a kernel module. Or even, we could imagine separating the cores in two halves, and somehow run (parts of) our real-time firmware side-by-side with Linux. In all cases, the goal is to make the measurement data available to userland programs on the RPi.

Your role will be to investigate all of these directions and get an understanding of the various barriers and bottlenecks involved. Depending on your pace of progress, we also have several intermediate milestones that would already be useful (e.g. adding features to the real-time firmware) as well as more advanced objectives (e.g. emulate an ultra-low-power energy harvester in real-time).

[1] `https://github.com/gberthou/rpi3-baremetal-template/tree/powerprobe`